邮政运输网络中的邮路规划和邮车调度问题

1 问题重述

古往今来，邮政在人们的生活中都扮演着不可或缺的角色。随着时代的发展，邮件投送的时限和成本成了邮政运输问题的关键因素。根据题目给出的实际情况，本文提出了关于如何合理规划邮路的问题，具体内容如下：

对一片有特定道路相连且有行政划分的地区进行邮路规划，有以下的问题需要解决：

(1) 以县局X_1及其所辖的16个支局$Z_1, Z_2, \cdots\cdots, Z_{16}$(下文简称为1，2，……)为研究对象。假设区级第一班次邮车08:00到达县局X_1，区级第二班次邮车16:00从县局X_1再出发返回地市局D，若每辆县级邮车最多容纳65袋邮件，在不超载的情况下，利用最少的车辆和最短的邮路，达到减少空车损失的目的。

(2) 采用尽可能少、尽可能短的邮路可以减少邮政部门车辆和人员等的投入，从而显著降低全区邮政运输网的总运行成本的邮路规划。

(3) 当县局可以跨县投寄时的邮路规划。

(4) 选择最合适的县局地点，并重新规划邮路，使得运行的成本最低。

2 模型假设

1．所有的邮车在邮路上均按照平均时速匀速行驶。

2．县局对市局送来邮件的集中处理时间(1小时)既包括区级邮车的装卸时间10分钟，也包括县级邮车的装卸时间10分钟。且在这1个小时的起始阶段进行装卸区级邮车的工作；而县级邮车的装卸工作最早在集中处理工作结束前10分钟进行，也可以在集中处理工作结束之后进行。

3．县局对将要送到市局的邮件的集中处理时间(1 小时)既包括县级邮车的装卸时间10分钟，也包括区级邮车的装卸时间10分钟。且在这1个小时的起始阶段进行装卸县级邮车的工作；而区级邮车的装卸工作最早在集中处理工作结束前10分钟进行，也可以在集中处理工作结束之后进行。

4．两班次的区级邮车行驶路线完全相同，若路线为环形则运行方向必须一致。如：D→61→58→53→X_5→52→59→60→D与D→60→59→52→X_5→53→58→61→D两种行车路线即为不同的两条路线。

5．问题4中选定县局后，县级邮车不得打破行政区划限制而跨县投寄。

3 符号说明

D：市级邮局

X_i：县级邮局

$X=\{X_1,X_2,X_3,X_4,X_5\}$：表示县级邮局的集合

$W(i,j)$：赋权邻接矩阵

$L(i,j)$：Floyd 算法中点 i 到 j 的距离。

$R(i,j)$：Floyd 算法中 i 到 j 之间的插入点。

$l(i.j)$：Floyd 算法中用插入顶点的方法依次构造出的距离矩阵。

$r(i,j)$：Floyd 算法中用插入顶点的方法依次构造出的路由矩阵。

$G=(V,E)$：表示无向图。

t_z：支局停留时间

t_X：县局停留时间

qs：区级邮车时速

t_{cl}：县局邮件集中处理时间

xs：县级邮车时速

T_i：区级邮车完成寄送县局 i 工作后返回市局所需要的时间

t_{i-j}：县级邮车在县 X_i 内走完第 j 条邮路所需要的时间

$Time_i$：开往县 X_i 的第一班次区级邮车开出市局与第二班次区级邮车到达市局所需要的时间。

$S(v_i)$：在各点 v_i 设立服务设施的最大服务距离

4 模型建立与求解

4.1 问题 1 的解决

4.1.1 模型的建立

根据题意，问题一可以归纳为如下数学模型。

$$\begin{cases}\min C(P_1,P_2,\cdots,P_k)\\ \min Lg(P_1,P_2,\cdots,P_k)\end{cases}$$
$$s.t.(P_1,P_2,\cdots,P_k)\in P$$

其中：$(P_1,P_2,\cdots,P_k)$表示邮路方案；$C(P_1,P_2,\cdots,P_k)$表示空置损失费；

$Lg(P_1,P_2,\cdots,P_k)$表示方案的总路径；P 表示邮路方案集。

4.1.2 方案的比较与确定

根据题目要求，需要在限定的时间内完成投送邮件的工作。首先，很自然地想到求出能够遍历这些点的最短路径，从理论上初步判断需要的车辆数。

4.1.2.1 Floyd 算法

Floyd 算法的基本思想就是直接在图的带权邻接矩阵中用插入顶点的方法依次构造出 v 个矩阵$L^{(1)}$、$L^{(2)}$、…、$L^{(v)}$，使最后得到的矩阵$L^{(v)}$成为图的距离矩阵，同时也求出插入点矩阵以便得到两点间的最短路径。

此算法的主要程序流程如下：

Step 1：输入赋权邻接矩阵$W(i,j)$，

Step 2：赋初值：对所有i，j，$l(i,j)\leftarrow W(i,j)$，$r(i,j)\leftarrow j$，$k\leftarrow 1$。

更新$l(i,j)$，$r(i,j)$：对所有i，j，若$l(i,k)+l(k,j)<l(i,j)$，则：

$l(i,j)\leftarrow l(i,k)+l(k,j),r(i,j)\leftarrow k$

Step 3：若$k=v$，停止，输出$L(i,j)=l(i,j)$、$R(i,j)=r(i,j)$；否则$k\leftarrow k+1$，重复 Step 1。

依照题目所给定数据得到的Floyd算法需要的赋权邻接矩阵如下：

0	27	44	17	11	27	42	inf	inf	inf	20	25	21	21	18	27	inf
27	0	31	27	49	inf	inf	inf	inf	inf	inf	inf	52	21	41	inf	inf
44	31	0	19	inf	27	32	inf	inf	inf	47	inf	inf	inf	50	inf	inf
17	27	19	0	14	inf	inf	inf	inf	inf	30	inf	inf	inf	31	inf	inf
11	49	inf	14	0	13	20	inf	inf	28	15	inf	inf	inf	15	25	30
27	inf	27	inf	13	0	9	21	inf	26	26	inf	inf	inf	28	29	inf
42	inf	32	inf	20	9	0	13	inf	32	inf	inf	inf	inf	inf	33	inf
inf	inf	inf	inf	inf	21	13	0	19	inf	inf	inf	inf	inf	inf	inf	inf
inf	inf	inf	inf	inf	inf	inf	19	0	11	20	inf	inf	inf	inf	33	21
inf	inf	inf	inf	28	26	32	inf	11	0	10	20	inf	inf	29	14	13
20	inf	47	30	15	26	inf	inf	20	10	0	18	inf	inf	14	9	20
25	inf	inf	inf	inf	inf	inf	inf	inf	20	18	0	23	inf	inf	14	inf
21	52	inf	inf	inf	inf	inf	inf	inf	inf	inf	23	0	27	22	inf	inf
21	21	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	27	0	inf	inf	inf
18	41	50	31	15	28	inf	inf	inf	29	14	inf	22	inf	0	11	inf
27	inf	inf	inf	25	29	33	inf	33	14	9	14	inf	inf	11	0	9
inf	inf	inf	inf	30	inf	inf	inf	21	13	20	inf	inf	inf	inf	9	0

（注：此矩阵中的 inf 表示邮局间无直达公路）　　具体程序见附件之程序一

4.1.2.2　TSP 算法

本文需要求解的每路邮车路线（区级或县级），若用顶点表示邮车经过的邮局（市局、县局或支局），边表示连接两邮局的公路，边上的权表示距离。实际我们的问题就是在加权图中寻找一条经过每个顶点至少一次的最短闭通路问题。

定义：在加权图 $G=(V,E)$ 中；

(1)权最小的哈密顿圈称为最佳 H 圈。

(2)经过每个顶点至少一次的权的闭通路称为最佳邮车路线。

最佳邮车路线问题可转化为最佳 H 圈问题，也称为 TSP 问题。

方法是由给顶的图 $G=(V,E)$ 构造一个以 V 为顶点集的完备图 $G'=(V,E')$，E'的每条边 (x,y) 的权等于顶点 x 与 y 在图中最短路的权,即：

$$\forall x,y\in E',\omega(x,y)=\min d_G(x,y)$$

定理：加权图 G 的最佳邮车路线的权与 G' 的最佳 H 圈的权相同。

算法：求加权图 $G=(V,E)$ 的最佳邮路的近似算法：

1．用 Floyd 算法求出 G 中任意两点间的最短路，构造出完备图 $G'=(V,E')$；

2．输入图 G' 的一个出始圈；

3．用算法产生一个初始 H 圈；

4．随机搜索出 G' 中若干个 H 圈，例如 20000 个；

5．对第 2、3、4 步所得的每个 H 圈，用二次逐次修正算法进行优化，得到近似最佳 H 圈；

6．在第 5 步求出的所有 H 圈中，找出权最小的一个，此即要找的最佳 H 圈的近似解。

具体程序见附件之程序二。

4.1.2.3　最少车辆的确定

根据题目的要求，寄达县局Z_i的邮件量为176袋，而收寄的邮件量为170袋，同时每一辆县级邮车最多容纳65袋邮件，因此至少需要出动3车次才能够完成这样的投送量。同时，当区级第一班次邮车08:00到达县局X_i之后需要有1个小时的时间对邮件进行集中处理；而当第二班次邮车16:00从县局X_i出发返回地市局D之前同样需要1个小时对邮件进行集中处理，因此县级邮车工作的时间为9:00到15:00之间的6个小时。

以下分情况讨论各种出车方案：

1．方案一：1辆车出动3次。

利用上面的Floyd算法和TSP算法联合求解，可以得到其最短里程数为279公里。以县级邮车平均时速30公里计算，1辆车至少需要9个多小时才可以完成，不满足6小时时限的条件。因此此本方案不可行。

2．方案二：2辆车各出动2次。

由算法得到本方案的最短里程为334公里，共需668分钟；再加上16个支局的装卸时间为80分钟，以及2辆车在县局的装卸时间20分钟，总共768分钟。因此平均每辆车耗时384分钟，超过了6小时时限。因此本方案也不可行。

3．方案三：2辆车，其中1辆车出动一次，另1辆车出动2次。

运用以上同样的方法可以得到其最短里程数为313公里，根据最高时速30公里的限制，可得需要626分钟；再加上16个支局的装卸时间为80分钟，以及其中一辆车在县局的装卸时间10分钟，共716分钟。因此平均每辆车耗时358分钟，恰好满足时间的限制。再考虑2辆车要送16个支局的因素，则至少有一辆车需要送8个支局。通过运行分组判断程序（程序三）可以得到的结论是：在6个小时的时间限制下，这样的邮路是不存在的。因此这种方案也不予考虑。

4．方案四：3辆车各出动1次。

由于出动的车次相同，本方案与方案二所需的最短里程数同为313公里，总共需时716分钟，平均每辆车耗时不到239分钟。由于有3辆车，则至少有一辆车需要投送6个支局。再运行分组判断程序则可以知道这样的邮路是存在的。因此本方案可行。

4.1.3　最佳邮路的选定方案

4.1.3.1　最佳邮路的确定

显然，本问题被转化为将16个支局分为3组，使得3辆从县局出发的车辆可以在6小时内到达自己组里的所有的支局并及时返回，同时在邮车运行过程中运载

的邮包不超过65袋且经济损失最小。对于运行过程中邮包的变化如表1所示：

表1：各支局邮件量及变化情况

支局 / 邮件量	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	Z_8	Z_9	Z_{10}	Z_{11}	Z_{12}	Z_{13}	Z_{14}	Z_{15}	Z_{16}
寄达局为Z_i的邮件量（袋）	10	15	6	9	13	6	11	4	13	17	11	2	11	21	13	14
支局Z_i收寄的邮件量（袋）	9	14	5	10	9	10	13	9	15	9	6	7	13	15	10	16
过支局Z_i邮件量的变化（袋）	-1	-1	-1	1	-4	4	2	5	2	-8	-5	5	2	-6	-3	2

根据题目所给的限制条件，引入贪心算法的概念。也就是尽量让邮车“吃饱”，即让空车率损失保持在较低的水平。最后结合 Floyd 算法重新编制改进型贪心算法（具体程序见附件程序四），其主要程序流程如下：

Step 1：将所有结点分为三组，并判断其运送的邮包量是否超过 65。如超过则重新分组，不超过则进入下一步。

Step 2：计算出 3 条遍历各分组节点的路径，并记录邮车每经过 1 个支局时总共损失的金额以及当时运载邮件的数量。判断运载邮件量是否超过 65，若超过则重复本步骤，不超过则进入下一步。

Step 3：判断得到的路径是否满足 6 小时的时间限制，若超过 6 小时则重复 Step 2，不超过则记录下损失金额并进入下一步。

Step 4：记录下 3 条路径的走法及损失的金额，并计算出路径总里程和总损失金额。

Step 5：重复执行 Step 2。若总损失金额大于先前所得，则重新执行 Step 2；若总损失金额小于先前所得，则覆盖原数据后重新执行 Step 2；若总损失金额等于先前所得，则判断总里程数，若小于先前所得则覆盖原数据重新执行 Step 2，否则直接重新执行 Step 2

Step 6：重新执行 Step 1，直到找到最小值。

利用以上的改进型贪心算法，得到邮路规划如表2所示：

表2：空车损失最小的邮路规划

邮路	耗时(分钟)	里程(公里)	损失(元)
X_1→15(不装卸)→16→15(不装卸)→5→6→2→3→4→X_1	348	159	5.108
X_1→12→13→1→14(不装卸)→15→14→X_1	325	150	19.292
X_1→10(不装卸)→9→8→7→8(不装卸) →9(不装卸)→11→10→X_1	321	148	22.615
		457(总计)	47.02(总计)

4.1.3.2　对最佳邮路的改进建议

以上方法得到的经济损失的数值固然最小，但是里程数与理论值313差距较大，特别是在邮车接近满载时程序会选择较长的路线，因为此时空车率极低（甚

至为0），所以即使路线较长但损失却不会过大。不过与现实生活中，运行里程也是要列入成本核算的，所以对上述的改进型贪心算法稍作修改：在Step 5中不再将所得数据与先前数据相比较，而是设定一个阈值（这里选定的是85），于是可以得到相对较短的3条邮路如表3：

表3：更符合实际的改进邮路

邮路	耗时(分钟)	里程(公里)	损失(元)
X_1→10(不装卸)→9→16→15→14→X_1	182	81	10.49
X_1→10→8→7→6→5→4→X_1	240	105	19.11
X_1→11→12→13→1→2→3→X_1	356	163	51.97
		349(总计)	81.57(总计)

4.1.4 问题 1 的小结

综上所述，经过建摸、编程、计算、求解等过程，我们得出至少需要 3 辆邮车才能满足该县的邮件运输需求。

为了降低空车率从而达到提高邮政运输效益的目的，根据题意给出具体的 3 条邮路规划如下：（具体的线路示意图见图 1）

1．X_1→15(不装卸)→16→15(不装卸)→5→6→2→3→4→X_1

2．X_1→12→13→1→14(不装卸)→15→14→X_1

3．X_1→10(不装卸)→9→8→7→8(不装卸) →9(不装卸)→11→10→X_1

这样的规划路线可使空车损失金额达到最小的 47.01 元，总里程为 457 公里。

同时，从实际出发，进一步给出一种兼顾空车损失金额和总里程数两方面因素的邮路规划如下：（具体的线路示意图见图 2）

1．X_1→10(不装卸)→9→16→15→14→X_1

2．X_1→10→8→7→6→5→4→X_1

3．X_1→11→12→13→1→2→3→X_1

这样的规划路线可使总里程数降为 349 公里，空车损失金额为 81.57 元，。

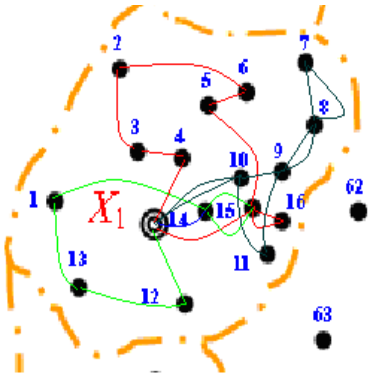

图1：空车损失最小的路线示意图

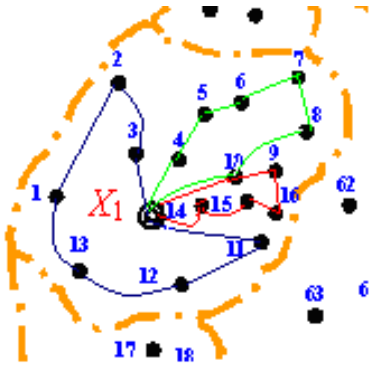

图2：兼顾两方面因素的路线示意图

4.2 问题 2 的解决

4.2.1 总体方案的选定及模型建立

要选择里程最短的路线首先应当考虑对整个区域采用环形邮路进行规划，但是由于有一整套严格的规定，这种方法显然是不可取的。而若采用辐射形邮路，根据题目所提供的信息便可以知道这种方法虽然能够满足时限要求，但所得路径的里程却很长。因此，混合形邮路就成了必然的选择。

问题二的数学模型如下：

$$\min \sum_{i,j,k} Y_{ijk} d_{ij}$$

$$s.t. \sum_{i} y_{ijk} = x_{jk}, j \in G \bigcup V, k \in K$$

$$\sum_{j} y_{ijk} = x_{ik}, i \in G \bigcup V, k \in K$$

$$\sum_{k} x_{jk} = 1, j \in V$$

$$\sum_{k} x_{jk} = m_i, j \in G = \{v_0, v_1, \cdots, v_5\}$$

$$X_{jk} = 0, j \in G_i, k \in k_s, l \neq s, s \neq 0$$

车辆集 K 满足时间约束

其中：

$k_i = \{1,2,\cdots,m_i\}, i = 0,1,\cdots,5$：表示区级及县级邮车集；$K = \bigcup k_i$。

$G_i = \{v_{i1}, v_{i2}, \cdots, v_{in_i}\}, i = 0,1,\cdots,5$：表示区级及县级支局集；$V = \bigcup G_i$。

$G = \{v_0, v_1, \cdots, v_5\}$，分别表示区级及县局集。

X_{ijk}为0、1变量，$i, j \in G \bigcup V, k \in \bigcup k_i$；$X_{ijk} = 1$，表示车辆 k 从结点 i 到结点 j。

Y_{ik}为0、1变量，$i \in G \bigcup V, k \in \bigcup k_i$；$Y_{ik} = 1$，表示车辆 k 服务于第 i 结点。

显然，此问题包括四个方面：第一、对各个邮局点进行合理分组；第二、在每组中求出最短回路；第三、判断是否满足时间约束；第四、选择总里程较短的线路方式。

4.2.2　具体方案的实施

4.2.2.1　以最小生成树为根据的邮路初步划分

由于单辆邮车的最佳路线问题不存在多项式时间内的精确算法，而图中节点数较多（总共为 79 个），且区级和县级邮车所覆盖的路线范围及路上所消耗时间还有诸多限制，因此我们只能去寻求一种较合理的划分准则对整个区域进行初步划分。

首先，分别列出从 D 点到 X_1、X_2、X_3、X_4、X_5 各支局的最短路，这些最短路构成一棵从 D 点出发的数枝，称为干枝。又因为各区级支局只能由区级邮车运送邮件，所以在此最小生成树图上需要加上所有的区级支局，并标示其到 D 点的最短路（见图 3）。

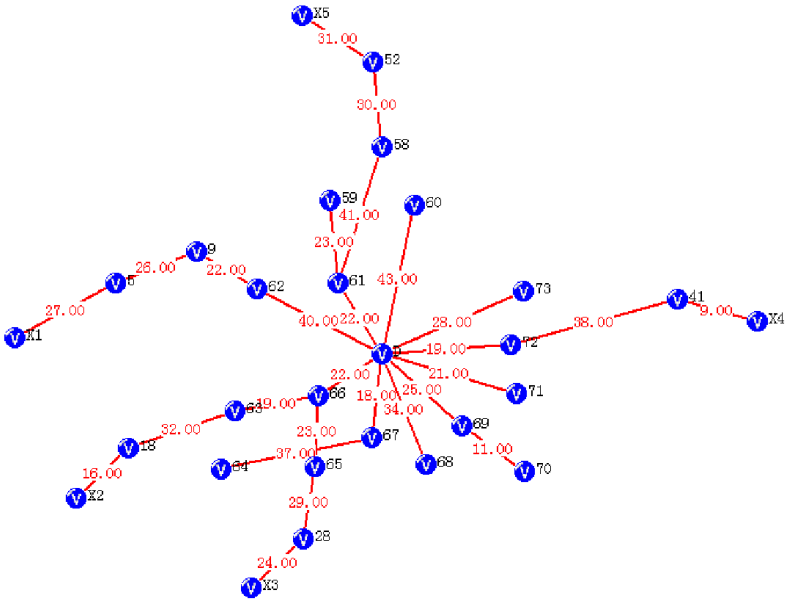

图 3：市局到各县局的最小生成树结构图

从图中可以看出，从 *D* 点出发到 5 个支局共有 5 条干枝，根据实际工作的经验及上述的分析，在分组时应遵循以下原则：

1．尽量将同一干枝上及其分枝上的节点分在同一组；

2．尽量将邻近并连通的节点分在同一组。

3．应让短的干枝和尽量多的与其邻近的节点分在同一组，而长的干枝和较少的与之接近的节点分在同一组。

对于剩下的各县级支局来说，根据其与市局及本县县局的关联程度，分别给它们赋予权值。最后依据上述分组原则，再使用 SPSS 软件求出以下的一种分组情况（见表 4）：

表 4：区级邮车到各县局所需要经过的支局

第一组	D，8，9，10，11，14，15，16，62，X_1
第二组	D，18，27，63，64，65，66，67，X_2
第三组	D，28，29，30，31，32，33，34，35，68，69，70，X_3
第四组	D，36，40，41，42，43，44，71，72，73，X_4
第五组	D，52，53，58，59，60，61，X_5

根据以上分组，使用先前的 TSP 算法程序可以分别得出每辆区级邮车运行的最佳路线。（见表 5）

表5：区级邮车的行驶路线

路线	里程数(公里)	耗　时(分钟)	到县局的耗时(分钟)	经停
D→62→16→15→11→X_1→14→10→9→8→62→D	226	259	105	县局X_1；支局62，16，15，11，14，10，9，8
D→66→63→64→18→X_2→27→65→67→D	232	259	125	县局 X_2；支局 66，63，64，18，27，65，67
D→68→29→33→X_3→28→31→30→32→34→35→70→69→D	249	295	114	县局X_3；支局68，29，33，28，31，30，32，34，35，70，69
D→71→36→41→X_4→40→43→44→42→73→72→D	248	284	84	县局X_4；支局71，36，41，40，43，44，42，73，72
D→61→58→53→X_5→52→59→60→D	267	286	133	县局X_5；支局61，58，53，52，59，60

（注：表中耗时依据假设1算定。）

4.2.2.2　以改进的 TSP 算法确定的邮路最终划分

接下来将原来的TSP算法程序稍加改动，为其加上时间限制条件。根据假设2可知县级邮车需在第一班次区级邮车到达县局之后60分钟后出发，但有10分钟可包含在这个60分钟内，因此对总消耗时间的影响为50分钟；根据假设3可知县级邮车需在第二班次区级邮车离开县局之前60分钟返回；又根据假设4规定的相同路线可知第一班次区级邮车到达县局所需的时间与第二班次区级邮车从县局返回市局所需的时间之和即为区级邮车完成寄送县局i工作后返回市局所需要的时间T_i。据此可以得到区级邮车最少总耗时 $Time_i = T_i + 50 + Max(t_{i-j}) + 60$ 。改进的TSP算法程序主要流程如下（具体程序见附件程序五）：

Step 1：求出县 X_1 中剩余支局间的最短路径，并解出 T_1、t_1 和 $Time_1$。若 T_1 大于300，则重新执行Step 1。

Step 2：判断 $Time_1$ 是否大于720。若小于720，则计算下一个县；若大于720，则将上述支局分为2组，重新计算 $Time_1$。判断 $Time_1$ 是否满足要求，若不满足则重新分组。如此循环

Step 3：若分2组不能满足要求就进一步将其分为3组，以此类推。

Step 4：按上述步骤将所有县级的邮路计算完毕。

各县内的具体邮路规划如表6：

表6：各县级邮车的行驶路线

路线	里程(公里)	耗时(分钟)	经停
X_1→1→13→12→X_1	96	207	支局1，13，12
X_1→4→5→7→6→2→3→X_1	126	282	支局 4，5，7，6，2，3
X_2→17→19→X_2	76	162	支局17，19
X_2→21→22→23→24→25→26→20→X_2	148	331	支局21，22，23，24，25，26， 20
X_4→37→38→39→X_4	92	199	支局37，38，39
X_5→54→57→56→55→54(不装卸)→X_5	132	284	支局54，57，56，55
X_5→50→49→48→47→45→46→51→X_5	137	309	支局50，49，48，47，45，46，51

4.2.2.2 具体的邮车调度方案

根据题意，可按以下步骤进行邮车调度：

Step 1：第一班次区级邮车早上 06:00 同时从市局出发开往县局。

Step 2：第一班次区级邮车到达各县局之后 10 分钟返回，县级邮车在区级邮车到达 1 小时后出发。

Step 3：最后一辆县级邮车回到县局的时间之后 1 小时就是第二班次区级邮车从该县局的发车时间，若此县无需县级邮车或县级邮车回县局时间较早，发车时间可相应延后。

Step 4：由第二班次区级邮车于各县局的发车时刻可倒推出区级邮车到达县局的时刻以及由市局出发的时刻，并检验此出发时刻是否晚于第一班次 区级邮车回到市局的时刻，若不相符合则调后第二班次区级邮车的发车时刻。

Step 5：根据以上的时刻信息计算出第二班次区级邮车返回到达市局的时刻，并检验是否超过 18:00，若超过 18:00 则重新调整。

最后得到邮车调度时刻表（见表7）如下：

表7：邮车调度时刻表

邮车	路线	出发时间	到达目的地时间	返回时间	返回到达时间
X_1(第一班)	D→62→16→15→11→X_1→14→10→9→8→62→D	06:00	07:45	07:55	10:19
X_1(第二班)		13:30	15:15	15:25(13:27)	17:49
X_2(第一班)	D→66→63→64→18→X_2→27→65→67→D	06:00	08:05	08:15	10:19
X_2(第二班)		13:21	15:26	15:36(14:36)	17:40
X_3(第一班)	D→68→29→33→X_3→28→31→30→32→34→35→70→69→D	06:00	07:54	08:04	10:55
X_3(第二班)		13:00	14:54	15:04(无)	17:55
X_4(第一班)	D→71→36→41→X_4→40→43→44→42→73→72→D	06:00	07:24	07:34	10:44
X_4(第二班)		13:00	14:24	14:34(11:43)	17:44
X_5(第一班)	D→61→58→53→X_5→52→59→60→D	06:00	08:13	08:23	10:46
X_5(第二班)		13:00	15:13	15:23(14:22)	17:46
X_{1-1}	X_1→1→13→12→X_1	08:45			12:12
X_{1-2}	X_1→4→5→7→6→2→3→X_1	08:45			13:27
X_{2-1}	X_2→17→19→X_2	09:05			12:14
X_{2-2}	X_2→21→22→23→24→25→26→20→X_2	09:05			14:36
X_{4-1}	X_4→37→38→39→X_4	08:24			11:43
X_{5-1}	X_5→50→49→48→47→45→46→51→X_5	09:13			14:22
X_{5-2}	X_5→54→57→56→55→54(不装卸)→X_5	09:13			13:57

（注：表中第二班次区级邮车返回时间栏中括号内的数值为该县县级邮车最晚的返回时间。）

4.2.3 问题 2 的小结

为了得到符合题目所提出的全部条件的路线规划结果，我们在这里以最小生成树为依据，先对邮路进行初步的划分，之后使用改进的TSP算法找出其余的线路。本线路规划方案规划邮路12条，需要区级邮车5辆，县级邮车7辆，共计12辆邮车；线路总里程2029公里，全区邮路运行成本6087元。

为了使本方案划分出的邮路得到更直观的表现，我们又制作了分县邮路表（表8）和邮路示意图（图4）。

表8：分县邮路表

区内	区级邮车路线	里程数（公里）	县内	县级邮车路线	里程数（公里）
X_1	D→62→16→15→11→X_1→14→10→9→8→62→D	226	X_{1-1}	X_1→1→13→12→X_1	96
			X_{1-2}	X_1→4→5→7→6→2→3→X_1	126
X_2	D→66→63→64→18→X_2→27→65→67→D	232	X_{2-1}	X_2→17→19→X_2	76
			X_{2-2}	X_2→21→22→23→24→25→26→20→X_2	148
X_3	D→68→29→33→X_3→28→31→30→32→34→35→70→69→D	249		无	
X_4	D→71→36→41→X_4→40→43→44→42→73→72→D	248	X_{4-1}	X_4→37→38→39→X_4	92
X_5	D→61→58→53→X_5→52→59→60→D	267	X_{5-1}	X_5→50→49→48→47→45→46→51→X_5	137
			X_{5-2}	X_5→54→57→56→55→54(不装卸)→X_5	132

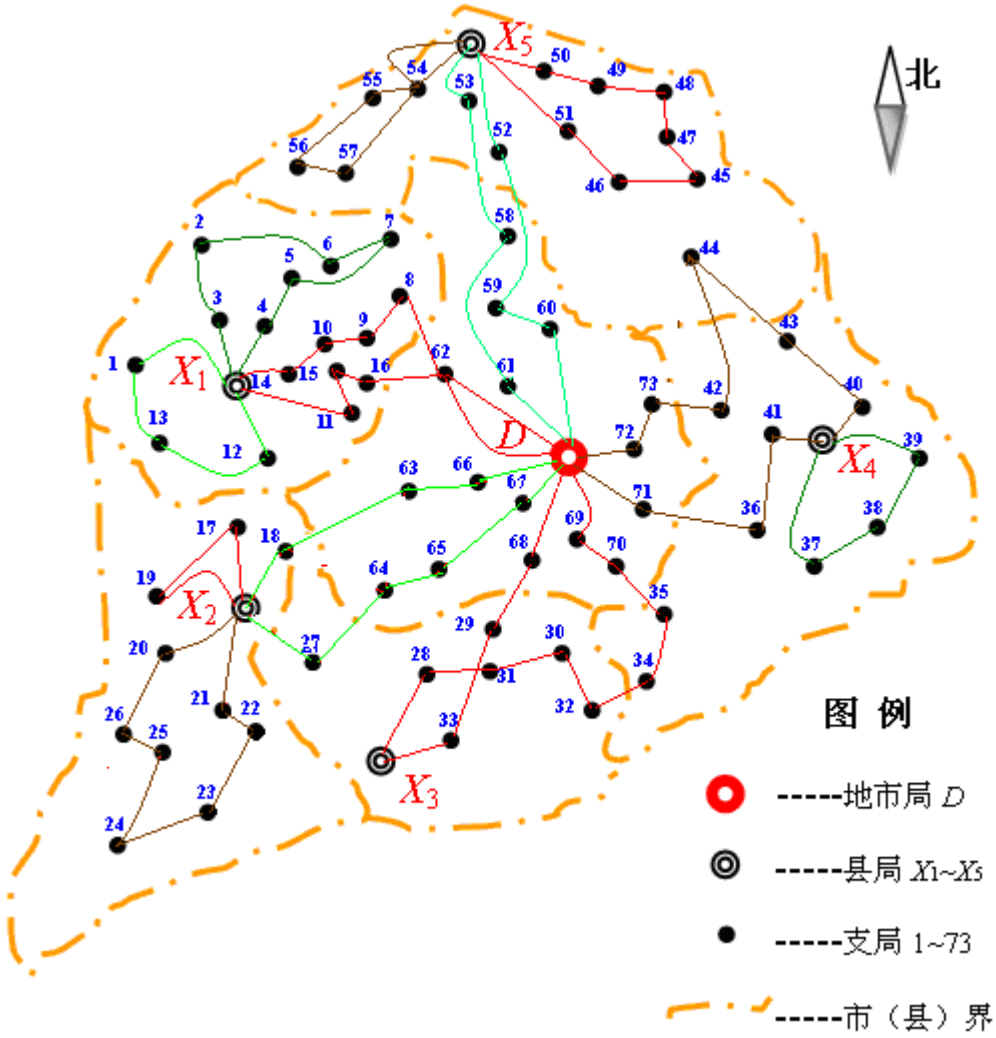

图4：全区邮路示意图

4.3 问题 3 的解决

4.3.1 解决方案的确定

对于打破行政区划的邮件投送，只需考虑部分县与县交界地带的支局到底由哪个支局进行投送即可。本文通过将这些支局分别放入相邻的两个县并计算最佳哈密顿圈（最佳 H 圈）的方法来确定支局的归属。即将此县级支局并入最佳哈密顿圈小的那个县。

4.3.2 具体方案的实施

4.3.2.1 并入邻县邮路支局的选择

根据问题 2 所得到的路线图，34 支局、35 支局和 44 支局都已经并入了邻县的邮路（由区级邮车完成投寄任务），只有 17 支局和 27 支局有并入邻县邮路的可能。

先考虑 17 支局的归属，当其没有并入相邻县局 X_1 的邮车运送范围时，县局 X_2 加支局 17～26 的最小哈密顿圈是 227 公里，县局 X_1 加支局 1～16 的最小哈密顿圈是 623；如果 17 支局并入相邻县局 X_1 的邮路时，县局 X_2 加支局 18～26 的最小哈密顿圈是 198，县局 X_1 加支局 1～17 的最小哈密顿圈是 630。（计算最小哈密顿圈的程序见附件之程序六。）再通过简单的计算有：227+623>198+630。可见，单从总里程数的减少上来说，17 支局可以划入 X1 邮车的运送范围。但是，进一步运用改进的 TSP 算法进行计算后发现，当有时间限制条件时所生成的邮路里程要大于原先的里程，因此把 17 支局划入 X1 内不可行。

再考虑 27 支局的归属，当其没有并入相邻县局 X_2 的邮车运送范围时，县局 X_3 加支局 27～33 的最小哈密顿圈是 176 公里，县局 X_2 加支局 17～26 的最小哈密顿圈是 227；如果 27 支局并入相邻县局 X_2 时 ，县局 X_3 加支局 28～33 的最小哈密顿圈是 134，县局 X_2 加支局 17～27 的最小哈密顿圈是 241。再通过简单的计算有：176+227>134+241。很明显，将 27 支局划入 X_2 邮车的运送范围能减少里程数，进一步运用改进的 TSP 算法进行计算后可以得到新的邮路规划方案使得里程数有所减少，因此把 27 支局划入 X2 内是可行的。

4.3.2.2 相关邮路的重新规划

通过以上程序的计算，可以得到新的邮路规划。即将原来D到X_2的路径D→66→63→64→18→X_2→27→65→67→D改为D→66→63→18→X_2→64→65→67→D，新邮路经停县局X_2和支局66，63，64，18，65，67，里程199公里，耗时224分钟，到X_2耗时97分钟。另外，将原先的邮路X_2→21→22→23→24→25→26→20→X_2改为X_2→27→21→22→23→24→25→26→20→X_2，经停支局27，21，22，23，24，26，25，20，里程169公里，耗时378分钟。

新的邮路规划方案还是需要 5 辆区级邮车和 7 辆县级邮车，共 12 辆车。总里程降低为 2017 公里，运行总成本为 6051 元。

4.3.3 问题 3 的小结

本方案通过对比最小哈密顿圈来确定将支局27划归邻县X_2的邮路以达到减少总里程数并降低成本的目的。相较于问题2的结果，总里程数降低了12公里，成本下降了36元。

为了使本方案划分出的邮路得到更直观的表现，我们又制作了邮车调度时刻表（表9）和邮路示意图（图5）。

表9：邮车调度时刻表

邮车	路线	里程数（公里）	出发时间	到达目的地时间	返回时间	返回到达时间
X_1（第一班）	D→62→16→15→11→X_1→14→10→9→8→62→D	226	06:00	07:45	07:55	10:19
X_1（第二班）			13:30	15:15	15:25	17:49
X_2（第一班）	D→66→63→18→X_2→64→65→67→D	199	06:00	07:37	07:47	09:44
X_2（第二班）			14:08	15:45	15:55	17:52
X_3（第一班）	D→68→29→33→X_3→28→31→30→32→34→35→70→69→D	249	06:00	07:54	08:04	10:55
X_3（第二班）			13:00	14:54	15:04	17:55
X_4（第一班）	D→71→36→41→X_4→40→43→44→42→73→72→D	248	06:00	07:24	07:34	10:44
X_4（第二班）			13:00	14:24	14:34	17:44
X_5（第一班）	D→61→58→53→X_5→52→59→60→D	267	06:00	08:13	08:23	10:46
X_5（第二班）			13:00	15:13	15:23	17:46
X_{1-1}	X_1→1→13→12→X_1	96	08:45			12:12
X_{1-2}	X_1→4→5→7→6→2→3→X_1	126	08:45			13:27
X_{2-1}	X_2→17→19→X_2	76	08:37			11:19
X_{2-2}	X_2→27→21→22→23→24→25→26→20→X_2	169	08:37			14:55
X_{4-1}	X_4→37→38→39→X_4	92	08:24			11:43
X_{5-1}	X_5→54→57→56→55→54→X_{55}	132	09:13			14:05
X_{5-2}	X_5→50→49→48→47→45→46→51→X_5	137	09:13			14:22

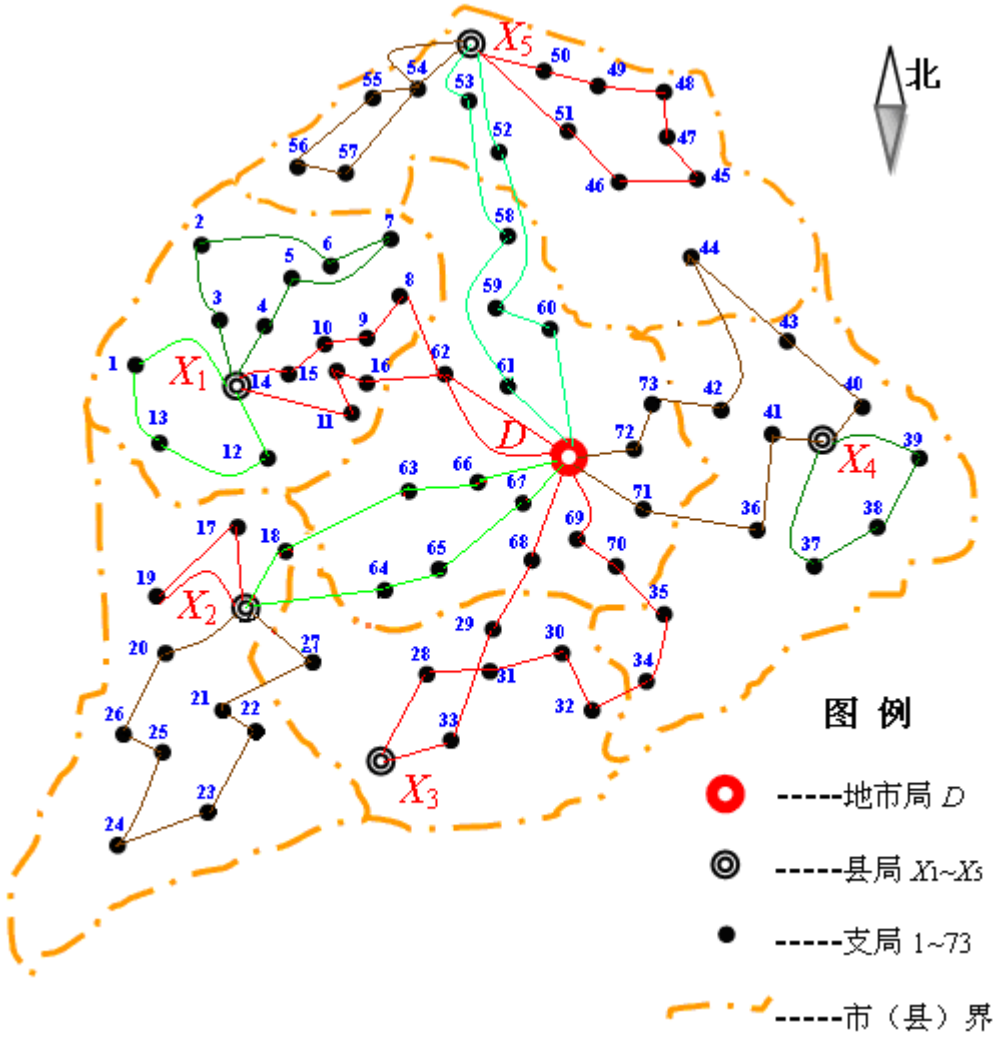

图5：邮路示意图

4.4 问题 4 的解决

4.4.1 解决方案的确定

很显然，本问题的关键在于县局地址选择的问题，县局地址一旦确定，根据原有的改进 TSP 算法就可以得到邮路的规划方案。

为了解决县局选址的问题，下面提出最短路覆盖中心算法（程序七），即要求网络中最边远的被服务点于提供公共服务的设施之间的距离尽可能小。具体的算法步骤如下：

Step 1：用 Floyd 算法求出距离矩阵 $L=\left(l_{ij}\right)_{v\times v}$；

Step 2：计算在各点 v_i 设立服务设施的最大服务距离 $S\left(v_i\right)$：

$$S\left(v_i\right)=\max_{1\le j\le v}\left\{d_{ij}\right\},i=1,2,\cdots,v$$

Step 3：求出顶点 v_k，使 $S\left(v_k\right)=\min_{1\le i\le v}\left|S\left(v_i\right)\right|$，则 v_k 技术要求的建立中心点的地点

4.4.2 方案的具体实施

4.4.2.1 各县级邮局选址的确定

利用上面提出的最短路覆盖中心算法，得到各县局的新位置如表10所示：

表10：新县局位置

县别	县局新
X_1	9
X_2	21
X_3	28
X_4	X_4
X_5	52

显然，X_4无需调整。对于其它新的县局地址的调整，依照假设5的限制条件，再调用改进的TSP算法进行演算，判断其可行性。最后得到以下结论：X_1、X_5调整后可减少里程，并符合时间限制，因此有调整的必要；X_3调整后的邮路与原先一致，所以没有调整的必要；X_2调整后虽然可以减少里程数，但不符合时间限制，因此也没有调整的必要。

4.4.2.2 重新调整部分县局位置后的邮路划分

通过上述的程序解得的与X_1有关的新邮路如下：

1. D→62→16→9(new X_1) →16→62→D

停县局 9(new X_1)，停支局 62，16
共 124 公里，耗时 135 分钟，到 X_5耗时 58 分钟

2．9(new X_1)→10→4→3→2→5→6→7→8→9(new X_1)
停支局 10，2，3，4，5，6，7，8
共 137 公里，耗时 314 分钟

3．9(new X_5)→11→12→13→1→X_1(old)→14→15→9(new X_1)
停支局 11，12，13，1，X_5(old)，14，15
共 161 公里，耗时 357 分钟

同时得到与 X_5有关的新邮路如下：

1．D→61→58→52(new X_5)→59→60→D
停县局 52(new X_5)，停支局 61，58，59，60
共 201 公里，耗时 216 分钟，到 X_5耗时 96 分钟

2．52(new X_5)→50→49→48→47→45→46→51→52(new X_5)
停支局 50，49，48，47，45，46，51
共 133 公里，耗时 301 分钟

3．52(new X_5)→53→X_5(old)→54→55→56→57→52(new X_5)
停支局 53，X_5(old)，54，55，56，57
共 164 公里，耗时 358 分钟

为了更直观地了解新邮路的分布情况，在此列出分县邮路表（见表11）：

表11：分县邮路表

邮车	区级邮车路线	里程数(公里)	邮车	县级邮车路线	里程数(公里)
X_1	D→62→16→9(new X_1)→16→62→D	124	X_{1-1}	9(new X_1)→10→4→3→2→5→6→7→8→9(new X_1)	137
			X_{1-2}	9(new X_5)→11→12→13→1→X_1(old)→14→15→9(new X_1)	161
X_2	D→66→63→64→18→X_2→27→65→67→D	232	X_{2-1}	X_2→17→19→X_2	76
			X_{2-2}	X_2→21→22→23→24→25→26→20→X_2	148
X_3	D→68→29→33→X_3→28→31→30→32→34→35→70→69→D	249		无	
X_4	D→71→36→41→X_4→40→43→44→42→73→72→D	248	X_{4-1}	X_4→37→38→39→X_4	92
X_5	D→61→58→52(new X_5)→59→60→D	201	X_{5-1}	52(new X_5)→50→49→48→47→45→46→51→52(new X_5)	133
			X_{5-2}	52(new X_5)→53→X_5(old)→54→55→56→57→52(new X_5)	164

经过调整X_1、X_5的县局位置，并重新规划邮路之后，得到的总里程数为1965公里，总成本为5895元。这个结果相对于问题2的结果，里程减少64公里，节约成本192元。

4.4.3 关于上报省局网运处的书面材料

尊敬的省局网运处领导：

为了全面贯彻中央关于全面建设和谐社会的方针；努力将邮政总局关于加强深化改革，保持平稳协调健康发展的措施落到实处；并结合本市的实际情况，提出关于本市邮政网络调整的一点建议，主要的理由如下：

1．本市邮政网络已运营多年，跟不上社会的发展和人民群众的需求，亟待调整和改革。

2．经测算，进行邮政网路调整可加快邮件投递速度并减少运营成本。例如，将县局X1迁至现在支局9的位置，同时将县局X5迁至现在支局52的位置，这样可减少运行里程64公里，并节约资金192元。

以上建议请批复，不周之处，敬请指正。

市局网运处

2007年10月22日

4.4.4 问题 4 的小结

通过使用最短路覆盖中心算法，首先将需要调整的县局迁移到合适的位置。运用原有程序得出邮路调整后的运行总里程数为1965公里，较问题2减少64公里；运行成本为5895元，较问题2减少支出192元。

为了更直观地展示调整后的邮路，在此编制邮车调度时刻表（表12）和邮路示意图（图6）。

表12：邮车调度时刻表

邮车	路线	里程数（公里）	出发时间	到达目的地时间	返回时间	返回到达时间
X_1（第一班）	D→62→16→9(new X_1)→16→62→D	124	06:00	06:58	07:08	08:15
X_1（第二班）			15:30	16:28	16:38	17:45
X_2（第一班）	D→66→63→64→18→X_2→27→65→67→D	232	06:00	08:05	08:15	10:19
X_2（第二班）			13:21	15:26	15:36	17:40
X_3（第一班）	D→68→29→33→X_3→28→31→30→32→34→35→70→69→D	249	06:00	07:54	08:04	10:55
X_3（第二班）			13:00	14:54	15:04	17:55

X_4（第一班）	D→71→36→41→X_4→40→43→44→42→73→72→D	248	06:00	07:24	07:34	10:44
X_4（第二班）			13:00	14:24	14:34	17:44
X_5（第一班）	D→61→58→52(new X_5)→59→60→D	201	06:00	07:36	07:46	09:36
X_5（第二班）			13:48	15:24	15:34	17:24
X_{1-1}	9(new X_1)→10→4→3→2→5→6→7→8→9(new X_1)	137	07:58			13:12
X_{1-2}	9(new X_5)→11→12→13→1→X_1(old)→14→15→9(new X_1)	161	07:58			13:55
X_{2-1}	X_2→17→19→X_2	76	09:05			12:14
X_{2-2}	X_2→21→22→23→24→25→26→20→X_2	148	09:05			14:36
X_{4-1}	X_4→37→38→39→X_4	92	08:24			11:43
X_{5-1}	52(new X_5)→50→49→48→47→45→46→51→52(new X_5)	133	08:36			13:37
X_{5-2}	52(new X_5)→53→X_5(old)→54→55→56→57→52(new X_5)	164	08:36			14:34

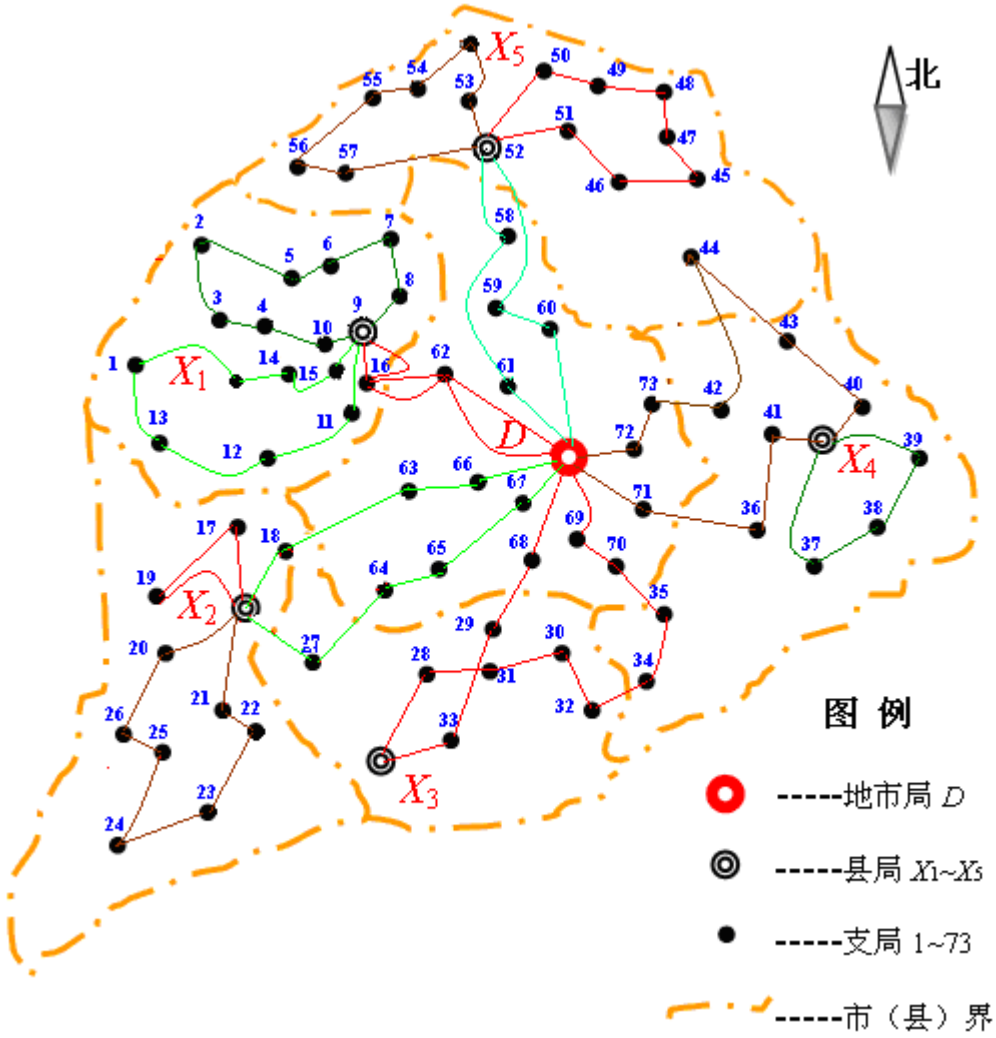

图6：邮路示意图

参考文献：

[1] 管梅谷.奇偶点图上作业法[J].数学学报,1990,10:263~266.
[2] 田丰,马仲蕃.图与网络流理论[M]。北京:科学出版社,2000.
[3] 吴振奎,刘舒强.运筹学概论[M]。第二版.北京:中国经济出版社,2000.
[4] Edmonds J, Johnson E L. Matching, Eulertouss and Chiness postman[J]. Math. Programming, 1973, 5: 88~124.
[5] 陈国良,等.遗传算法及其应用[M].北京:人民邮电出版社,1999.4~160.
[6] 戴一奇,等.图论与数据结构[M].北京:清华大学出版社,1998.254~309.
[7] 杨弼亮.汽车运输企业管理基础[M].北京:人民交通出版社,1992.123~225.
[8] 胡运权.运筹学教程.北京:清华大学出版社,1998.225~228,129~136,241~242.
[9] Robert V Binder 著.华庆一等译.面向对象系统的测试[M].北京:人民邮电出版社,2001.
[10] C Bourhr,R Dssouli,E M Aboulhamid.Automatic test generation for EFSM-based systems[M].Publication departementale 1043,Departement IRO,Universite de Montreal,1996-08.
[11] cavalla A R,Favreau J-P,Philippou M.Formal Methods in Conformance Testing:Results and Perspectives[C].In:Procedings of Protocol Test Systems VI,IFIP,1994:3~17.
[12] A V Aho,A T Dahbura,D Lee et al.An optimization technique for protocol conformance test generation based on UIO sequences and rural Chinese postman tours[J].IEEE trans Commum,1991,39(11):1604~1615.
[13] G Prem Kumar,P Venkataram.Protocol Test Sequence Generation Using MUIOS Based on TSP Problem [C].In:National Conference on Computer Networks,Architecture and Applications,Madras,1994-12:65~74
[14] 王治平,李雪耀.遗传算法求解有向中国邮局问题[J].哈尔滨工程大学学报,1998;19(2).
[15] Syam Siddhartha S.A model and methodologies for the location problem with logistical components[J].Computers & Operations Research,2002,29:1173~1193.
[16] Kaj Holmberg.Exact solution methods for uncapacitated location problems with convex transportation costs[J].European Journal of Operationl Research,1999,114:127~140.

附件:

程序一

```
function WT1_floyd
%Floyd 算法得出的最短距离矩阵和路径矩阵，可求任意几点或所有点。
D=ones(79,79)*inf;%赋权邻接矩阵
for i=1:79
    D(i,i)=0;
end
fid=fopen('sysj1.txt');
tline = fscanf(fid,'%5d');
fclose(fid);
[L,CM]=size(tline);
for i=1:L/3
     %i
     a=tline(3*(i-1)+1);
     b=tline(3*(i-1)+2);
     c=tline(3*i);
     if a<7
         x=a;
     elseif a<20
         x=a-10+6;
     else
         x=a-100+6;
     end
     if b<7
         y=b;
     elseif b<20
         y=b-10+6;
     else
         y=b-100+6;
     end
     D(x,y)=c;
 end
 for i=2:79
     for j=1:(i-1)
         D(i,j)=D(j,i);
     end
 end
 %求任意几点
% a=[2+6,3+6,4+6,5+6,6+6,7+6];
```

```
% n=length(a);
% W1=zeros(n,n);
 %for i=1:n
  %   for j=1:n
   %      W1(i,j)=D(a(i),a(j));
   % end
   %end
 [d,r]=Floyd(D);
 %d
 %[d,r]=Floyd(W1);
```

程序二

```
sets:
  cities/1..3/:u;
  link(cities, cities):  distance,    x;
endsets

data:
distance=1  2  3
4  5  6
7  8  9;


!distance = @OLE('D:\LINGO讲稿\tsp2.xls');

enddata
n=@size(cities);
min=@sum(link(i,j)|i #ne# j: distance(i,j)*x(i,j));
@for(cities(i) :
   @sum(cities(j)| j #ne# i: x(j,i))=1;
   @sum(cities(j)| j #ne# i: x(i,j))=1;
   @for(cities(j)| j #gt# 1 #and# j #ne# i :
       u(i)-u(j) +n* x(i,j)<=n-1;
   );
);
! Make the x's 0/1;
@for(link : @bin(x));
! For the first and last stop;
@for(cities(i) | i #gt# 1 :
   u(i)<=n-1-(n-2)*x(1,i);
   u(i)>=1+(n-2)*x(i,1);
);
```

程序三

```
% search three best paths for  length and time
%最后程序报错，则表明从未有分组满足条件，1 量车不能在规定时间内跑完 X1 的 8 个支局
function WT1_zxkn
a=[10   15  6   9   13  6   11  4   13  17  11  2   11  21  13  14];


c=0;c1=0;c2=0;
bestcost=100000;
y=zhs(16,8);

for i=1:size(y,1)

    b=1:16;
  s1=sum(a(y(i,:)));
  c1=c1+1;
    if (s1<=65)& (176-s1<=130)
     b(y(i,:))=[];

      p1=y(i,:);p2=b; % 分组方案
      time1=cost2(p1);
      if time1<=360


          c=c+1
          result(c,:)=p1;

  end
end
end
result
```

程序四

```
function WT1

a=[10   15  6   9   13  6   11  4   13  17  11  2   11  21  13  14];

fenzu=[];
c=0;
bestcost=100;
```

```
for k=6:6
y=zhs(16,k);

for i=1:size(y,1)

     b=1:16;
  s1=sum(a(y(i,:)));
    if (s1<=65)& (176-s1<=130)
     b(y(i,:))=[];
     for j=4:(16-k)/2


         z=zhs(16-k,j);
         for ii=1:size(z,1)
             b1=b(z(ii,:));
             s2=sum(a(b1));
             if (s2<=65) & (176-s1-s2<=65)
                 c=c+1  ;
      p1=y(i,:);p2=b1;b2=b;b2(z(ii,:))=[];p3=b2; % 分组方案
      [best1,bestcost1]=WT1_value(p1);          [best2,bestcost2]=WT1_value(p2);
[best3,bestcost3]=WT1_value(p3);
      if (bestcost1+bestcost2+bestcost3<bestcost)
          bestcost=bestcost1+bestcost2+bestcost3
          bestpath=[best1(2:end)-1,best2(2:end)-1,best3(2:end)-1];
          bestno=[size(p1,2),size(p2,2),size(p3,2)];
          bestp=[p1,p2,p3];
      end
 end
end
end



    end

end
end
```

程序五

```
function zuijzh
%改进的 TSP 算法，即加入了时间限定的分组,由于组合较多，程序运行时间较长
a=[4,8,9,10,11,14,15,16,62,79];%D 车到达县 Xi 的最短路径,及 D 车到 Xi 必须经过的点。
c=[1,2,3,5,6,7,12,13];
%以下是对区级邮车所经支局的组合优化选择算法
```

```
%n=length(a);
%m=length(c);
%t1_max=300;
%t1_min=0;
%for i=2
 %   i
    %加入 i 个新点到 D-X
  %  dz=zeros(1,n+i);
    %dz(1:n)=a;
    %knzh=nchoosek(c,i);
    %[p,q]=size(knzh);
    %for j=1:p
    %    dz(n+1:n+i)=knzh(j,:);
     %   [length1,path]=hamiton(dz);
      %  t1=5*(n+i)+(length1*60)/65;
       % if t1_min<t1<t1_max
        %    t1_min=t1;
         %   t1_p=path;
          %  l1=length1;
           %end
           %end
    %dz
    %end
for i=4
    i
    %加入 i 个新点到 D-X
    dz1=zeros(1,i+1);
    dz1(1,1)=4;
    dz2=zeros(1,length(c)-i+1);
    dz2(1,1)=4;
    knzh=nchoosek(c,i);
    [p,q]=size(knzh);
    %p
    km=2;
    L_min=197;
    for j=1:p
        j
        dz1(2:i+1)=knzh(j,:);
        for k=1:length(c)
            for kn=1:j
                if c(k)~=knzh(j,kn)
                    dz2(km)=c(k);
                    km=km+1;
                end
```

```
            end
        end
        [length1,path1]=hmt(dz1);
        [length2,path2]=hmt(dz2);
        t1=5*i+length1*2;
        t2=5*(length(c)-i)+length2*2;
        if t1>t2
            t=t1+(224*60)/65;
            if t<610
                L=224+length1+length2;
                if L<L_min
                    L_min=L;
                    path=path1;
                end
            end
        else
            t=t2+(224*60)/65;
            if t<610
                L=224+length1+length2;
                if L<L_min
                    L_min=L;
                    path=path2;
                end
            end
        end
    end
    %dz
End
```

程序六

```
function [length,path]=hmt(dian)
%dian=[1,2,3];
% 输入节点编号 输出 hamiton 圈及长度
fid=fopen('hmtd.txt');
tline = fscanf(fid,'%5d');
fclose(fid);
%[L,CM]=size(tline);
d=zeros(79,79);
for i=1:79
    for j=1:79
        d(i,j)=tline((i-1)*79+j);
    end
end
```

```
n=size(dian,2)*size(dian,1);
xx=1:n;
length=0; path=dian(xx);
for i=1:n-1
    length=length+d(dian(xx(i)),dian(xx(i+1)));
end
length=length+d(dian(xx(n)),dian(xx(1)));
while (1)
        i=n;
        while (xx(i-1)>=xx(i))
            i=i-1;
        end
            k=n;
        while (xx(i-1)>=xx(k))
            k=k-1;
        end
                  temp= xx(i-1);xx(i-1)=xx(k); xx(k)=temp;
         for j=i:i+(n-i)/2
         temp=xx(j);xx(j)=xx(n-j+i);xx(n-j+i)=temp;
     end
        length1=0;
     for i=1:n-1
    length1=length1+d(dian(xx(i)),dian(xx(i+1)));
end
length1=length1+d(dian(xx(n)),dian(xx(1)));
     if length1<length
         length=length1;
         path=dian(xx);
     end

        flag=0;
        for j=2:n
            if (xx(j)~=n-j+2)
                flag=1;break;
            end
        end
         if (flag==0)
             break;
         end

     end

      %length
      %path
```

程序七

```
function zxd
D=ones(79,79)*inf;
for i=1:79
    D(i,i)=0;
end
fid=fopen('sysj1.txt');
tline = fscanf(fid,'%5d');
fclose(fid);
[L,K]=size(tline);
for i=1:L/3
     %i
     a=tline(3*(i-1)+1);
     b=tline(3*(i-1)+2);
     c=tline(3*i);
     if a<7
         x=a;
     elseif a<20
         x=a-10+6;
     else
         x=a-100+6;
     end
     if b<7
         y=b;
     elseif b<20
         y=b-10+6;
     else
         y=b-100+6;
     end
     D(x,y)=c;
 end
 for i=2:79
     for j=1:(i-1)
         D(i,j)=D(j,i);
     end
 end
 dian=[4,27+6,28+6,29+6,30+6,31+6,32+6,33+6];
 [cg,Lg]=size(dian);
 W1=zeros(Lg,Lg);
 for i=1:Lg
     for j=1:Lg
         W1(i,j)=D(dian(i),dian(j));
```

```
    end
end
[d,r]=Floyd(W1);
min=inf;
zxd=0;
for i=1:Lg
    s(i)=max(d(i,:));
    if s(i)<min
        zxd=i;
        min=s(i);
    end
end
zxd
```